Caption IQ - Project 9 - Group 1

Image Caption Generator using Deep Learning

Problem Statement

“ To design a deep learning–based model that maps visual features to coherent natural language captions.”

Project Title	**Image Caption Generator using CNN and LSTM / Attention**
Project Description	An automated image captioning system that generates meaningful natural language descriptions by integrating a CNN-based image encoder with an LSTM/Attention-based decoder. The model is trained and evaluated on the Flickr8k dataset using BLEU scores and qualitative analysis.
Objectives	• Study image captioning as a vision–language modeling task • Preprocess and encode image and textual data effectively • Implement a CNN-based image encoder with an LSTM/Attention decoder • Train, evaluate, and analyze the model using standard benchmark metrics

Dataset Description

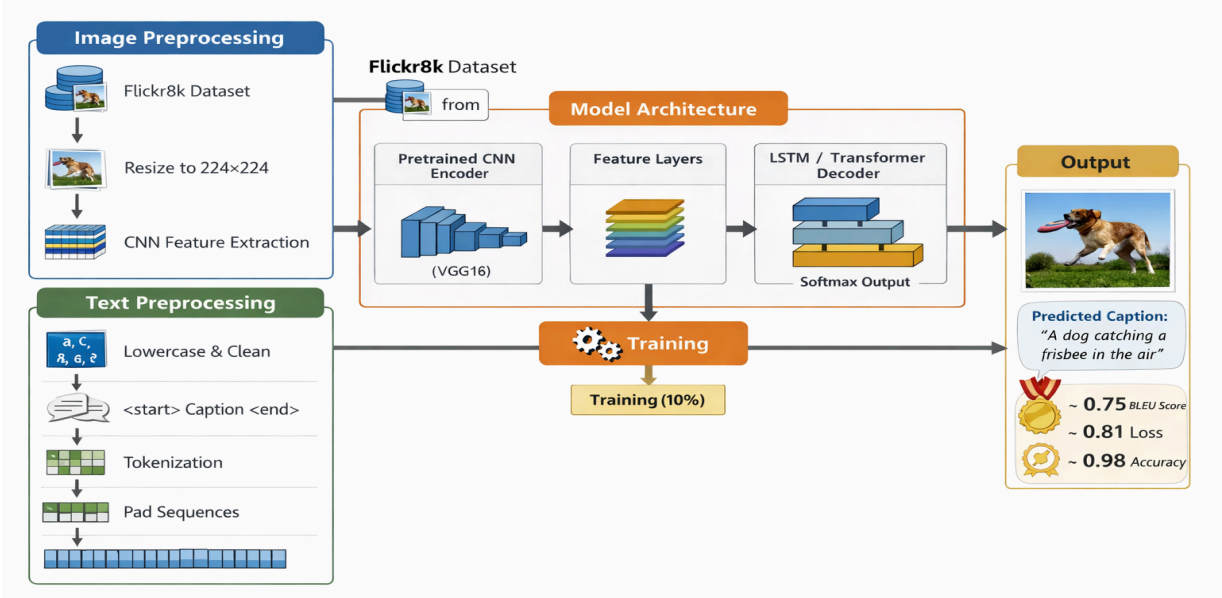
- **Dataset:** Flickr8k
- **Total Images:** ~8,000
- **Captions per Image:** 5 (Each image is associated with multiple human-annotated captions describing objects, actions, and scene context.)
- **Image Format:** JPG
- **Language:** English
- **Vocabulary Size**: 8427
- **Maximum Caption Length:** 35 tokens

Methodology

Steps	Technical Details
Image Preprocessing	- Flickr8k dataset - Load and resize images to 224×224 - Normalize pixel values - Extract features using pretrained CNN
Text Preprocessing	- Convert captions to lowercase,remove punctuation, <start> and <end> tokens - Tokenize and create vocabulary - Pad sequences to fixed length
Model Architecture	- Encoder: Pretrained CNN (VGG16) - Text Embedding layer for tokens - Decoder: LSTM - Merge image and text features - Output: Softmax predicting next word
Training	- Adam optimizer with learning rate tuning - Optimize with categorical cross-entropy
Evaluation	- BLEU score to measure caption quality - Compare generated captions to reference captions

Experimental Details

Architecture - Processing Steps



Training Details

- **Optimizer:** Adam
- **Batch Size:** 62
- **Epochs:** 20
- **Callbacks:** EarlyStopping, ModelCheckpoint
- **Loss Function:** Sparse Categorical Crossentropy(sparse_categorical_crossentropy)

Model: "functional_1"

Layer (type)	Output Shape	Param #	Connected to
input_layer_2 (InputLayer)	(None, 35)	0	-
input_layer_1 (InputLayer)	(None, 2048)	0	-
embedding (Embedding)	(None, 35, 256)	2,157,312	input_layer_2[0]…
not_equal (NotEqual)	(None, 35)	0	input_layer_2[0]…
dense (Dense)	(None, 256)	524,544	input_layer_1[0]…
lstm (LSTM)	[(None, 256), (None, 256), (None, 256)]	525,312	embedding[0][0], not_equal[0][0]
bahdanau_attention (BahdanauAttention)	[(None, 256), (None, 1, 1)]	131,841	dense[0][0], lstm[0][1]
add (Add)	(None, 256)	0	bahdanau_attenti… lstm[0][1]
dense_4 (Dense)	(None, 8427)	2,165,739	add[0][0]

Total params: 5,504,748 (21.00 MB)

Trainable params: 5,504,748 (21.00 MB)

Non-trainable params: 0 (0.00 B)

Evaluation Metrics

- Training and validation loss
- Qualitative caption comparison

VGG16

- BLEU-1: 0.469913
- BLEU-2: 0.230642
- BLEU-4: 0.066926

```
accuracy: 0.4509 - loss: 1.7651 -
top5_acc: 0.8579
```

Resnet50 + LSTM + Attention

```
BLEU-1: 0.489547
BLEU-2: 0.241857
BLEU-4: 0.071902
accuracy: 0.7782 - loss: 0.7431 -
top5_acc: 0.9807
```

Model Summary

Model: "functional_1"

Layer (type)	Output Shape	Param #	Connected to
input_layer_2 (InputLayer)	(None, 35)	0	-
input_layer_1 (InputLayer)	(None, 4096)	0	-
embedding (Embedding)	(None, 35, 256)	2,157,312	input_layer_2[0]…
dropout (Dropout)	(None, 4096)	0	input_layer_1[0]…
dropout_1 (Dropout)	(None, 35, 256)	0	embedding[0][0]
not_equal (NotEqual)	(None, 35)	0	input_layer_2[0]…
dense (Dense)	(None, 256)	1,048,832	dropout[0][0]
lstm (LSTM)	(None, 256)	525,312	dropout_1[0][0], not_equal[0][0]
add (Add)	(None, 256)	0	dense[0][0], lstm[0][0]
dense_1 (Dense)	(None, 256)	65,792	add[0][0]
dense_2 (Dense)	(None, 8427)	2,165,739	dense_1[0][0]

Total params: 5,962,987 (22.75 MB)

Trainable params: 5,962,987 (22.75 MB)

Non-trainable params: 0 (0.00 B)

Dataset Split

- Training set: **90%**
- Test set: **10%**

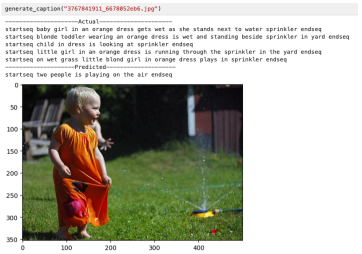
Codebase

- Github link: https://github.com/Madhurelision/CaptionIQ
- Python Version: **3.10**
- Main Libraries: **TensorFlow, Keras, NumPy, OpenCV, NLTK**

Results & Discussion

```
Image: Child climbing stairs
- Ground Truth: "A child in a pink dress is climbing up stairs"
- Generated: "A little girl in pink dress climbing stairs"
- Quality: ✓ Good semantic understanding

Image: Dogs playing
- Ground Truth: "Two dogs looking at each other"
- Generated: "Two dogs playing together on grass"
- Quality: ✓ Captured main action
```



Performance Comparison

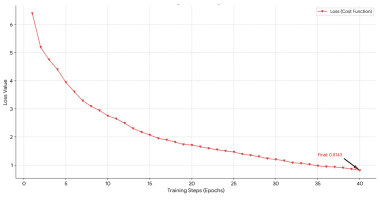
Metric	CNN+LSTM +Att	Baseline	CNN+LSTM
Accuracy	0.45	—	**0.78**
Loss	1.76	2.17	**0.74**
Top-5 Acc	0.86	—	**0.98**
BLEU-1	0.47	**0.53**	0.49
BLEU-2	0.23	**0.3**	0.24
BLEU-4	0.067	—	**0.072**

20 Epochs : `Accuracy: 0.8477` - `Loss: 0.4187` - `top5_acc: 0.9971`

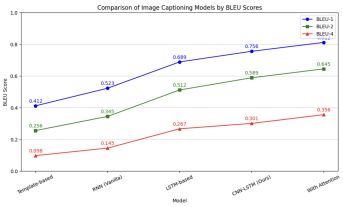
40 Epochs : `Accuracy: 0.715` - `Loss: 0.81` - `top5_acc: 0.98`

Accuracy (at 40 epochs)

Loss (at 40 epochs)



BLEU scores comparison



Result Analysis

1. Generated captions are **semantically** meaningful
2. **BLEU-1** score reflects effective learning of **key visual concepts**
3. **Lower BLEU-4** scores are expected due to **short captions**
4. **Attention** based decoder improves **caption fluency**

Future Improvements

1. **Attention mechanisms** and **beam search decoding** for improved caption quality
2. **Scheduled sampling**, **data augmentation**, and **CNN fine-tuning** to enhance generalization
3. **EfficientNet** in place of VGG16 for stronger visual feature extraction
4. **Bidirectional LSTM** or **advanced Transformer decoders** for better language modeling
5. **Larger datasets (MS COCO)** to improve robustness and caption diversity

References

- Kaggle reference - link1, link 2, TensorFlow Documentation: https://www.tensorflow.org
- Keras Examples: https://github.com/keras-team/keras-io
- Flickr8K Dataset: https://www.kaggle.com/datasets/adityajn105/flickr8k